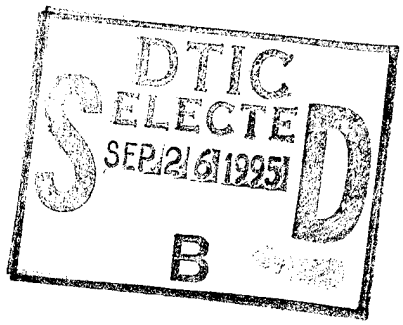
Management Science Research Report Number #602

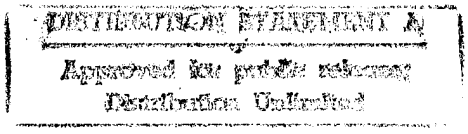
Polyhedral Methods for the Maximum Clique Problem

Egon Balas
Sebastián Ceria
Gérard Cornuéjols
Gábor Pataki



February 1994

Graduate School of Industrial Administration
Carnegie Mellon University
Pittsburgh, PA 15213



This research was support in part by a National Science Foundation Grant Number DDM-9201340 and the Office of Naval Research grant N00014-89-J-1063.

Management Science Research Group
Graduate School of Industrial Administration
Carnegie Mellon University
Pittsburgh, PA 15213

Polyhedral Methods for the Maximum Clique Problem

Abstract

This paper presents an integer programming approach to the maximum clique problem. An initial linear programming relaxation is solved and, when there is an integrality gap, this relaxation is strengthened using one of several tightening procedures. This is done through the addition of cutting planes to the linear program. The bulk of the paper deals with theoretical and computational issues associated with the generation of these cuts. In particular, we describe how to obtain cuts from the positive semi-definiteness of an underlying matrix. The various cuts are then compared in a computational experiment. These cuts can be incorporated into a branch-and-cut algorithm and we report results with such an algorithm on some of the DIMACS benchmark instances.

1 Introduction

A *clique* in a graph $G = (V, E)$ is a vertex set $C \subseteq V$ such that every two vertices of C are adjacent. The *clique polytope* $C(G)$ is the convex hull of the incidence vectors of the cliques in G:

$$C(G) = conv\{x \in \{0,1\}^n : x_i + x_j \leq 1, \forall (i,j) \notin E\},$$

where $n = |V|$. The maximum weight clique problem consists of finding $max\{wx : x \in C(G)\}$. The linear programming relaxation of $C(G)$

$$FC(G) = \{x \in \Re^n_+ : x_i + x_j \leq 1, \forall (i,j) \notin E\}$$

is sometimes called the *fractional clique polytope*. This polytope strictly contains $C(G)$ whenever G is not the complement of a bipartite graph, and there are several well known classes of inequalities which are valid for $C(G)$ but violated by points in $FC(G)$. The addition of some of these inequalities to $FC(G)$ gives rise to stronger relaxations of $C(G)$. A vertex set S is *stable* if every two vertices in S are nonadjacent. Clearly, for any stable set S of size greater than two, $\sum_{i \in S} x_i \leq 1$ is valid for $C(G)$ and strengthens the inequalities $x_i + x_j \leq 1$ of $FC(G)$ for all $i, j \in S$.

Our approach in this paper is to strengthen the relaxation $FC(G)$. We start with a very simple strengthening,

$$SC(G) = \{x \in \Re^n_+ : \sum_{i \in S_k} x_i \leq 1, \text{ for } k = 1, \ldots m\}$$

where the sets S_k, $k = 1, \ldots, m$, are maximal stable sets with the property that, for any $(i,j) \notin E$, there exists at least one k such that $i, j \in S_k$. Constructing such a family S_k, $k = 1, \ldots, m$, is done using a straightforward greedy procedure. The focus of the paper is on procedures that achieve further strengthenings in a systematic fashion. We illustrate how some recent results on lift-and-project tightening procedures for mixed 0-1 programs can be applied to relaxations of the clique polytope such as $SC(G)$.

In Section 2 we describe three different lift-and-project tightening procedures: the first one is the procedure of Balas, Ceria and Cornuéjols [1] and is the simplest of the three, while the other two can be interpreted as variations of tightening procedures of Lovász and Schrijver [6]. In all three cases the problem is formulated in a higher dimensional space, where the resulting relaxation can be tightened by imposing the 0–1 condition on one or more variables. The tightened formulation is then projected onto the space of original variables.

In Section 3, we show how to generate cutting planes that arise from these tightening procedures. We can generate the "deepest" cut of the family by solving a linear program.

In Section 4, we explain how the higher dimensional formulations can be further strengthened by requiring positive semidefiniteness of a certain matrix defined by the newly introduced variables. This idea is also related to a tightening procedure of Lovász and Schrijver [6], different from those mentioned above.

In Section 5, we show that finding a deepest cut for the positive semidefinite cuts requires the solution of a semi-infinite linear program where the entering column is found by an eigenvalue computation.

Finally, in Section 6, we report on our computational experience. The first part of the computational study investigates the quality of the cuts obtained from the various tightening procedures. In the second part, we apply MIPO to some of the DIMACS benchmark instances: MIPO is a general purpose mixed integer program optimizer based on the tightening procedure of [1].

2 Lift-and-project tightening procedures

Before we can discuss the algorithms used in our computational study to strengthen relaxations of the maximum clique problem in the x-space of vertex variables, we need to lift the problem into a higher dimensional space. Indeed, the tightening procedures are easily understood in that higher space. This section discusses several options for the tightening step. A projection step is then used to return to the x-space. We delay until Section 3 the discussion of algorithms for finding a deepest cut from this projection.

Let K be a relaxation of the clique polytope $C(G)$ such that $C(G) = conv(K^0)$, where $K^0 = \{x \in \{0,1\}^n : x \in K\}$. We represent the inequalities defining K by the system $Ax \geq b$.

2.1 The basic lift-and-project procedure

The following results are from Balas, Ceria and Cornuéjols [1], and are also valid for more general pure and mixed 0–1 programs. Consider a fixed $j \in \{1, \ldots, n\}$.

Procedure 1

1. *Multiply $Ax \geq b$ with $1 - x_j$ and x_j to obtain the nonlinear system*

$$\begin{aligned} (1 - x_j)(Ax - b) &\geq 0 \\ x_j(Ax - b) &\geq 0. \end{aligned} \tag{1}$$

2. *Linearize (1) by substituting y_i for $x_i x_j$, $i = 1, \ldots, n$, $i \neq j$, and x_j for x_j^2. Call the polyhedron defined by the resulting system $M_j(K)$.*

3. *Project $M_j(K)$ onto the x-space. Call the resulting polyhedron $P_j(K)$.*

The calculation of the projection used in Step 3 amounts to eliminating the variables y_i by taking certain specific linear combinations of the inequalities defining $M_j(K)$. Even though the system $M_j(K)$ is small (the number of its nonzero entries is of order n), computing the full projection $P_j(K)$ is computationally prohibitive in general. In Section 3, we will show how to obtain useful inequalities defining $P_j(K)$.

It is not hard to see that $K^0 \subseteq P_j(K) \subseteq K$. In fact, $P_j(K)$ has the following considerably stronger properties:

Theorem 2.1 $P_j(K) = conv(K \cap \{x : x_j \in \{0,1\}\})$.

For $t \geq 2$, define $P_{i_1,\ldots,i_t}(K) = P_{i_t}(P_{i_1,\ldots,i_{t-1}}(K))$.

Theorem 2.2 *For any $t \in \{1, \ldots, n\}$,*
$P_{i_1,\ldots,i_t}(K) = conv(K \cap \{x : x_j \in \{0,1\}$ *for* $j = i_1, \ldots, i_t\})$.

Corollary 2.3 $P_{1,\ldots,n}(K) = conv(K^0)$.

Theorem 2.2 implies the following.

Property 2.4 *For $i \neq j$,*
$conv(conv(K \cap \{x : x_i \in \{0,1\}\}) \cap \{x : x_j \in \{0,1\}\})$
$= conv(conv(K \cap \{x : x_j \in \{0,1\}\}) \cap \{x : x_i \in \{0,1\}\})$
$= conv(K \cap \{x : x_i, x_j \in \{0,1\}\})$.

Theorem 2.2 says that if Procedure 1 is applied repeatedly, each time with respect to a different variable x_j, for $j \in S$, the outcome is $conv\{x \in K : x_j \in \{0,1\} \text{ for } j \in S\}$. In other words, the ordering of $i_1, \ldots, i_t$ in the definition of $P_{i_1,\ldots,i_t}(K)$ is irrelevant. This justifies the simpler notation $P_S(K)$. A further result (see Sherali and Adams [7] and also [1]) states that if Step 1 of Procedure 1 is applied repeatedly with respect to several variables x_j, for $j \in S$, followed by a single linearization and projection, the outcome is the same as if Procedure 1 had been applied repeatedly in its entirety. To be specific, consider the following procedure for a given $S \subseteq \{1, \ldots, n\}$.

Procedure 1'

1. *Multiply* $Ax \geq b$ *with* $\prod_{i \in S_1} x_i \prod_{j \in S_2}(1 - x_j)$ *for every partition* (S_1, S_2) *of* S, *to obtain the nonlinear system*

$$\prod_{i \in S_1} x_i \prod_{j \in S_2} (1 - x_j)\,(Ax - b) \geq 0, \quad \forall S_1, S_2 \subseteq S \text{ s.t. } S_1 \cup S_2 = S,\ S_1 \cap S_2 = \emptyset. \quad (2)$$

2. *Linearize (2) by substituting* x_j *for* x_j^p, $p \geq 2$, *and* y_W *for* $\prod_{i \in W} x_i$, $|W| \geq 2$. *Call the polyhedron defined by the resulting system* $M'_S(K)$.

3. *Project* $M'_S(K)$ *onto the* x*-space. Call the resulting polyhedron* $P'_S(K)$.

We then have

Theorem 2.5 $P'_S(K) = P_S(K)$.

For proofs of these theorems, as well as their connection with earlier work and further details, the reader is referred to [1].

2.2 Stronger tightening procedures

In this section, we use other inequalities (different from $(1 - x_j) \geq 0$ and $x_j \geq 0$) in the multiplication step of lift-and-project tightening procedures. This idea was introduced by Lovász and Schrijver [6]. For a detailed description of this and other lift-and-project tightening procedures the reader is referred to Ceria [4].

Given any stable set S, the inequality $x(S) \leq 1$ is valid for $C(G)$. The modified lift-and-project tightening procedure is defined as follows:

Procedure 2

1. *Multiply* $Ax \geq b$ *with* $1 - x(S)$ *and* x_i, $i \in S$, *to obtain the nonlinear system*

$$\begin{aligned} (1 - x(S))(Ax - b) &\geq 0 \\ x_i(Ax - b) &\geq 0, \quad i \in S. \end{aligned} \tag{3}$$

2. *Linearize (3) by substituting* $y_{i\ell}$ *for* $x_i x_\ell$, $\ell = 1, \ldots, n$, $i \in S$, *and* x_i *for* x_i^2, $i \in S$, *and setting* $y_{i\ell} = y_{\ell i}$. *Call the polyhedron defined by the resulting system* $H_S(K)$.

3. *Project* $H_S(K)$ *onto the* x*-space. Call the resulting polyhedron* $R_S(K)$.

Appying Procedure 2 amounts to convexifying over *all* the variables in S as shown by the following theorem.

Theorem 2.6 $R_S(K) = conv(K \cap \{x : x_i \in \{0,1\},\ for\ all\ i \in S\})$.

Proof: We claim that applying Step 1 of Procedure 2, then carrying out the multiplications in (3) and substituting x_i for x_i^2, $i \in S$, yields the same nonlinear system, say $(*)$, that we obtain by applying Step 1 of Procedure 1' with respect to S, carrying out the multiplications in (2) and substituting x_i for x_i^p, $p \geq 2$. Then the theorem follows since the linearization and projection of $(*)$ onto the x-space yields $conv(K \cap \{x : x_i \in \{0,1\}, \forall i \in S\})$ by Theorem 2.5.

To prove the claim, notice that after carrying out the multiplications in (2) and substituting x_i for x_i^p, $i \in S$, $p \geq 2$, all those terms containing the product $x_i x_j$ for some pair $i, j \in S$ become 0 since, for each such pair, $1 - x_i - x_j \geq 0$ is part of the system $Ax - b \geq 0$, and $x_i(1 - x_i - x_j) \geq 0$ implies $-x_i x_j \geq 0$. Thus the only inequalities of $(*)$ with nonzero terms are

- the inequality corresponding to the partition (S_1, S_2) with $S_1 = \emptyset$ and $S_2 = S$ which is

$$Ax - b - \sum_{i \in S}(Axx_i - bx_i) \geq 0.$$

- the $|S|$ inequalities corresponding to partitions of the form (S_1, S_2) with $S_1 = \{i\}$, $S_2 = S \setminus \{i\}$, which are

$$Axx_i - bx_i \geq 0,\ for\ i \in S.$$

But these are precisely the inequalities obtained by executing the multiplications in (3), which proves the claim. □

Let T be a set of vertices of G and let $\mathcal{S} = \{S_1, \ldots, S_k\}$ be a cover of the vertices of T by stable sets $S_j \subseteq T$. When $\mathcal{S}$ is a partition, this is a k-coloring of $G(T)$, but in general we will not choose $\mathcal{S}$ to be a partition. For $2 \leq t \leq k$, define $R_{S_1,\ldots,S_t}(K) = R_{S_t}(R_{S_1,\ldots,S_{t-1}}(K))$.

Theorem 2.7 *For any* $t \in \{1, \ldots, k\}$,
$R_{S_1,\ldots,S_k}(K) = conv(K \cap \{x : x_i \in \{0,1\}, \forall i \in S_1 \cup \ldots \cup S_t\})$.

Proof: By induction on t. For $t = 1$, the statement is just Theorem 2.6 applied to $S = S_1$. Suppose the statement is true for $t = 1, \ldots, p-1$ and let $t = p$. Then

$$\begin{aligned}
R_{S_1,\ldots,S_p}(K) &= R_{S_p}(R_{S_1,\ldots,S_{p-1}}(K)) && (4)\\
&= R_{S_p}(conv(K \cap \{x : x_i \in \{0,1\},\ \forall i \in S_1 \cup \ldots \cup S_{p-1}\})) && (5)\\
&= conv(conv(K \cap \{x : x_i \in \{0,1\},\ i \in S_1 \cup \ldots \cup S_{p-1}\})\\
&\quad \cap\{x : x_i \in \{0,1\},\ \forall i \in S_p\}) && (6)\\
&= conv(K \cap \{x : x_i \in \{0,1\},\ \forall i \in S_1 \cup \ldots \cup S_p\}) && (7)
\end{aligned}$$

where (4) follows from the definition, (5) from the induction hypothesis, (6) from Theorem 2.6 and (7) from Property 2.4. □

Corollary 2.8 $R_{S_1,\ldots,S_k}(K) = conv(K \cap \{x_i \in \{0,1\}, i \in T\})$.

Proof: Follows directly from Theorem 2.7. □

Applying the previous result with $T = V$ implies that we can obtain $C(G)$ in a number of steps equal to the cardinality of any stable set cover of the vertices of G. Of course, this number is bounded below by the chromatic number of G. In a random graph with edge probability of $1/2$, one can obtain a stable set cover of size in the order of $n/\log_2 n$ with probability $1 - o(1)$.

It is possible to further generalize these results by allowing for multiplication with inequalities corresponding to more than one stable set in $\mathcal{S}$. In particular, we are interested in the following extension:

Procedure 3

1. *Multiply* $Ax \geq b$ *with* $1 - x(S_j)$, $j = 1, \ldots, k$ *and* x_i, $i \in T$ *to obtain the nonlinear system*

$$\begin{aligned} (1 - x(S_j))(Ax - b) &\geq 0, \quad j = 1, \ldots, k \\ x_i(Ax - b) &\geq 0, \quad i \in T. \end{aligned} \tag{8}$$

2. *Linearize (8) by substituting* $y_{i\ell}$ *for* $x_i x_\ell$, $\ell = 1, \ldots, n$, $i \in T$, *and* x_i *for* x_i^2, $i \in T$, *and setting* $y_{i\ell} = y_{\ell i}$. *Call the polyhedron defined by the resulting system* $H_{1\ldots k}(K)$.

3. *Project* $H_{1\ldots k}(K)$ *onto the* x*-space. Call the resulting polyhedron* $R_{1\ldots k}(K)$.

The special case where $T = V$ and each S_j reduces to a single vertex of G, say $S_j = \{j\}$, is one of the tightening procedures proposed by Lovász and Schrijver [6]. When $T \neq V$ and $S_j = \{j\}$ for $j \in T$, let us denote by $N_T(K)$ the polyhedron obtained in Step 3 of Procedure 3. It is easy to see that, for any $\mathcal{S}$,

$$R_{1\ldots k}(K) \subseteq N_T(K)$$

and, in fact, the relaxation $R_{1\ldots k}(K)$ can be substantially stronger when some of the S_j's in $\mathcal{S}$ have cardinality greater than one. For example, when $K = FC(G)$, Theorem 2.6 implies that

$$R_{1\ldots k}(K) \subseteq \cap_{j=1}^{k} conv(K \cap \{x : x_i \in \{0,1\} \text{ for all } i \in S_j\})$$

whereas it can be shown that

$$N_T(K) = \cap_{j \in T} conv(K \cap \{x : x_j \in \{0,1\}\}).$$

3 Lift-and-Project Cutting Planes

In this section we address the generation of cutting planes for the maximum clique problem

$$max\{wx : \; x \in K, \, x_j \in \{0,1\}, \, j = 1, \ldots, n\}, \tag{9}$$

where, as before, $K = FC(G)$ or any valid relaxation of $C(G)$. The cut generation algorithms presented in this paper work indifferently for the maximum cardinality or maximum weight clique problems.

3.1 Cut generation linear program for Procedure 1

First, we consider Procedure 1 discussed in Section 2. Balas, Ceria and Cornuéjols [1] showed that in order to generate a valid inequality $\alpha x \geq \beta$ for $P_j(K)$ that cuts off $\bar{x}$ (the current solution to the linear programming relaxation of (9)) by the largest amount, where $0 < \bar{x}_j < 1$, we need to solve the following linear program:

$$\begin{aligned} & max\ uA\bar{x} + u_0\bar{x}_j \\ & \text{subject to} \\ & uA - vA + (u_0 - v_0)e_j = 0 \\ & ub = -1 \\ & vb + v_0 = -1 \\ & u, v \geq 0. \end{aligned} \tag{10}$$

Substituting $v_0 = -vb - 1$ into $uA_j - vA_j + u_0 - v_0 = 0$, and then substituting

$$\begin{aligned} u_0 &= -uA_j + vA_j - vb - 1 \\ &= -uA_j + vA_j + ub - vb \end{aligned}$$

into the objective function, yields the linear program

$$\begin{aligned} & max\ -u((b - A_j)\bar{x}_j + A\bar{x}) + v(b - A_j)\bar{x}_j \\ & \text{subject to} \\ & (-u + v)A_i = 0 \quad i \neq j \\ & ub = -1 \\ & u, v \geq 0, \end{aligned} \tag{11}$$

where A_i denotes the i^{th} column of A. Then the cut $\alpha x \geq \beta$ is obtained as follows:

$$\begin{aligned} \alpha_i &= uA_i \qquad i \neq j \\ \alpha_j &= ub + v(A_j - b) \\ \beta &= ub = -1. \end{aligned}$$

3.2 Cut generation linear programs for Procedures 2 and 3: an illustration

Now we discuss the generation of cutting planes for the stronger tightening procedures of Section 2. They require solving linear programs whose size depends on the sets S or T. We illustrate the cases where S is a stable set of size 2 (Procedure 2) and T is a clique of size 2 (Procedure 3).

Suppose $S = \{j, \ell\}$, where $(j, \ell) \notin E$ in Procedure 2. A deepest cut (i.e. an inequality $\alpha x \geq \beta$ valid for $R_S(K)$ which cuts off $\bar{x}$ by the largest amount) is obtained by solving the linear program:

$$
\begin{array}{l}
max \ -u((b-A_j)\bar{x}_j + A\bar{x}) + v^j(b-A_j)\bar{x}_j - u((b-A_\ell)\bar{x}_\ell + A\bar{x}) + v^\ell(b-A_\ell)\bar{x}_\ell \\
\text{subject to} \\
\begin{array}{lrcll}
(-u+v^j)A_\ell & +(-u+v^\ell)A_j & = & 0 & \\
(-u+v^j)A_i & & = & 0 & i \neq j, \ell \\
 & (-u+v^\ell)A_i & = & 0 & i \neq j, \ell \\
ub & & = & -1 & \\
 & u, v^j, v^\ell & \geq & 0 &
\end{array}
\end{array}
\tag{12}
$$

and defining

$$
\begin{array}{rcll}
\alpha_i & = & uA_i & i \neq j, \ell \\
\alpha_j & = & ub + v^j(A_j - b) & \\
\alpha_\ell & = & ub + v^\ell(A_\ell - b) & \\
\beta & = & ub = -1. &
\end{array}
$$

The linear programs (11) and (12) optimize over the sets $conv(K \cap \{x : x_j \in \{0,1\}\})$ and $conv(K \cap \{x : x_j, x_\ell \in \{0,1\}\})$, respectively. The corresponding inequalities $\alpha x \geq \beta$ belong to the family of disjunctive cuts, to which the strengthening procedure of Balas and Jeroslow [3] can be applied.

Now let $T = \{j, \ell\}$, where $(j, \ell) \in E$. We consider Procedure 3 where $S_1 = \{j\}$ and $S_2 = \{\ell\}$. The deepest cut $\alpha x \geq \beta$ is obtained by solving the linear program:

$$
\begin{array}{lll}
max & -u^j((b-A_j)\bar{x}_j + A\bar{x}) + v^j(b-A_j)\bar{x}_j - u^\ell((b-A_\ell)\bar{x}_\ell + A\bar{x}) + v^\ell(b-A_\ell)\bar{x}_\ell & \\
\text{subject to} & & \\
(-u^j+v^j)A_\ell & +(-u^\ell+v^\ell)A_j = 0 & \\
(-u^j+v^j)A_i & = 0 \quad i \neq j,\ell & \\
 & (-u^\ell+v^\ell)A_i = 0 \quad i \neq j,\ell & \\
u^jb & +u^\ell b = -2 & \\
 & u^j, v^j, u^\ell, v^\ell \geq 0 &
\end{array}
\tag{13}
$$

and defining

$$
\begin{aligned}
\alpha_i &= (u^j + u^\ell)A_i \qquad i \neq j,\ell \\
\alpha_j &= (u^j - v^j)b + u^\ell A_j \\
\alpha_\ell &= (u^\ell - v^\ell)b + u^j A_\ell \\
\beta &= u^jb + u^\ell b = -2.
\end{aligned}
$$

The linear program (13) is significantly larger than (11) but it is closely related to it and this relationship can be used to avoid solving (13) from scratch. Indeed, assuming that we have solved (11) for two different fractional components of $\bar{x}$, say j and ℓ, we can combine the resulting optimum solutions to obtain a feasible solution to (13). Then, using sensitivity analysis, one can improve upon the combined solution by noting that the difference between linear program (13) and the two versions of (11) is two pairs of aggregated constaints, namely $(-u^j + v^j)A_\ell = 0$ and $(-u^\ell + v^\ell)A_j = 0$ on the one hand, and $u^jb = -1$ and $u^\ell b = -1$ on the other hand. Increasing the RHS of one constraint in the pair by δ in one linear program and decreasing it by δ in the other will improve the solution of (13) at a rate given by the difference of the corresponding dual variables in (11). The updated u^j, v^j, u^ℓ, v^ℓ yield a stronger cut in general, although not necessarily the strongest for (13).

The linear programs (12) and (13), just like (11), can be restricted to the space of fractional components in $\bar{x}$, and the resulting cuts can be lifted, as explained in [1].

3.3 Cut generation linear program for $N_T(K)$

In this section, we give the linear program that needs to be solved for finding a deepest cut which is valid for $N_T(K)$. Remember that $N_T(K)$ arises from Procedure 3 by imposing the disjunctions $\vee_{j\in T}(x_j = 0 \vee x_j = 1)$.

We start with the higher dimensional relaxation that arises in Step 2 of Procedure 3. It contains the constraints

$$\begin{array}{rrrrrcl} x & -z^j & & -y^j & & = & 0 \\ & Az^j & -bz_0^j & & & \geq & 0 \\ & z_j^j & & & & = & 0 \\ & & & Ay^j & -by_0^j & \geq & 0 \\ & & & y_j^j & -y_0^j & = & 0 \\ & & z_0^j & & +y_0^j & = & 1 \end{array}$$

for all $j \in T$, together with the symmetry constraints

$$y_k^j - y_j^k = 0$$

for all $j, k \in T$ such that $j < k$. Any valid inequality $\alpha x \geq \beta$ for $N_T(K)$ is obtained from this system by eliminating the y and z variables. This yields:

$$\begin{array}{rrrrrcll} \sum_{j\in T}\alpha^j & & & & & = & \alpha & \\ \alpha^j & -u^jA & -u_0^j e_j & & & = & 0, & \forall j \in T \\ \alpha^j & -v^jA & -v_0^j e_j & -\sum_{k>j} w_{jk}e_k + \sum_{\ell<j} w_{\ell j}e_\ell & & = & 0, & \forall j \in T \\ & u^jb & & & & \geq & \beta_j, & \forall j \in T \\ & v^jb & +v_0^j & & & \geq & \beta_j, & \forall j \in T \\ & & & & \sum_{j\in T}\beta_j & \geq & \beta & \\ & & & & u^j,\ v^j & \geq & 0. & \end{array}$$

If we eliminate the α^j's, we can write the cut generation linear program for finding the deepest valid cut for $N_T(K)$ as follows.

$$
\begin{aligned}
Max \quad & \sum_{j\in T} u^j(-A\bar{x}) + \sum_{j\in T} u_0^j(-\bar{x}_j) \\
(u^j - v^j)A + (u_0^j - v_0^j)e_j - \sum_{k>j} w_{jk}e_k + \sum_{\ell<j} w_{\ell j}e_\ell &= 0, \quad \forall j \in T \\
u^j b &\geq \beta_j, \ \forall j \in T \\
v^j b + v_0^j &\geq \beta_j, \ \forall j \in T \\
\sum_{j\in T} \beta_j &\geq \beta \\
u^j, \ v^j &\geq 0.
\end{aligned}
$$

Note that it is possible to solve this linear program by restricting it to the subspace corresponding to the fractional components of $\bar{x}$. Indeed, it can be shown, using an argument similar to that in [1], that the solution obtained in this subspace can be lifted *optimally* into a deepest cut for the full space.

4 Positive Semi-Definite Constraints

In this section and the next, we show how to further strengthen the polytope $N_T(K)$. For ease of notation, we present the results for $T = V$. Once again, we first derive the results in the higher dimensional space.

The idea of using a positive semi-definite (psd for short) constraint in higher dimensional formulations of 0-1 integer programs appeared first in [6] : if in Step 2 of Procedures 2 or 3 all the $x_i x_\ell$ products appear $1 \leq i, \ell \leq n$, then it is a valid constraint to enforce psdness of the symmetric matrix Y substituted for xx^T. In the case where $K = FC(G)$, the higher dimensional relaxation $M(K)$ can be substantially strengthened by psdness. In this section, we show how to construct a closely related family of constraints.

We denote psdness of a matrix A by writing $A \succeq 0$. Let Y be the symmetric matrix of our variables $y_{i\ell}$, diag $Y = x$, Y_0 some $n \times p$ matrix formed from elements of Y, $p \leq n$, and $S \succeq 0$ some fixed $p \times p$ matrix. We consider the following *psd constraint*

$$
Y - Y_0 S Y_0^T \succeq 0. \tag{14}
$$

The *separation problem* for (14) is to check whether (14) holds for given matrix Y and, if it does not, to produce a cutting plane which is valid for all matrices satisfying (14) but not

for the given matrix. In the following we show that (14) behaves well algorithmically and we give two choices for S and Y_0 with the property that the resulting constraint (14) is valid for our higher dimensional representations.

Theorem 4.1 *The separation problem for (14) can be solved in polynomial time.*

Proof. Suppose we are given Y and let Y_0 be some $n \times p$ matrix formed from elements of Y, $p \leq n$. Since $S \succeq 0$ we can write $S = Z^T Z$ for some matrix Z. Then Y satisfies (14) if and only if

$$u^T(Y - Y_0 S Y_0^T)u \ \geq 0 \quad \text{for all } u \in \Re^n,\ |u| = 1 \tag{15}$$

i.e.

$$Y \circ uu^T - |ZY_0u|^2 \ \geq 0 \quad \text{for all } u \in \Re^n,\ |u| = 1 \tag{16}$$

where $\circ$ denotes the inner product of matrices. For a moment, let us fix u, and let $a = ZY_0u$. An elementary calculation shows that

$$-|a|^2 = \min\{2\lambda^T a + |\lambda|^2 : \lambda \in \Re^n\} \tag{17}$$

where the minimum is attained for $\lambda = -a$. Thus (16) is equivalent to

$$\min\{Y \circ uu^T + 2\lambda^T(ZY_0u) + |\lambda|^2 : \lambda \in \Re^n\} \ \geq 0 \quad \text{for all } u \in \Re^n,\ |u| = 1 \tag{18}$$

which is the same as

$$Y \circ uu^T + 2\lambda^T(ZY_0u) + |\lambda|^2 \ \geq 0 \quad \text{for all } u \in \Re^n,\ |u| = 1, \text{for all } \lambda \in \Re^n. \tag{19}$$

Thus, (19) is a linearized form of (14), and separation goes as follows:

1. Find $u \in \Re^n, |u| = 1$, s.t. $u^T(Y - Y_0 S Y_0^T)u < 0$. If no such u exists, $Y - Y_0 S Y_0^T \succeq 0$. This step is of order n^3 since it amounts to performing Gaussian elimination, see [5] page 295.

2. If u exists, find λ s.t. $2\lambda^T(ZY_0u) + |\lambda|^2 = |(ZY_0u)|^2$ (i.e. $\lambda = -(ZY_0u)$). Then Y violates (19) with the above λ and u.

■

Our first choice for S and Y_0 which makes (14) valid for the higher dimensional relaxations is $S = I$, $Y_0 = x$. Thus we obtain the constraint

$$Y - xx^T \succeq 0 \tag{20}$$

This constraint can be expressed in a different way, related to a construction of Lovász and Schrijver.

Proposition 4.2 *Let Y be given, diag Y=x,*

$$Y_* = \begin{pmatrix} 1 & x \\ x & Y \end{pmatrix}$$

Then $Y - xx^T \succeq 0$ if and only if $Y_* \succeq 0$.

Proof. Define the n by n matrix L as

$$L = \begin{pmatrix} 1 & 0 \\ -x & I \end{pmatrix}$$

Since L is of full rank, clearly Y_* is psd if and only if LY_*L^T is psd. However, it is easy to check that

$$LY_*L^T = \begin{pmatrix} 1 & 0 \\ 0 & Y - xx^T \end{pmatrix},$$

thus the proposition follows. ■

The matrix Y_* is related to the work of Lovász and Schrijver: in [6], they homogenize their lower dimensional formulation $Ax \geq b$ into $Ax - bx_0 \geq 0$. Thus by multiplying variables they obtain an $n+1$ by $n+1$ matrix having its first row and column equal to its diagonal. The top left element in the matrix is $x_0 = 1$. Imposing psdness on this matrix thus is equivalent to our constraint (20).

Our second choice for S and Y_0 is $S = (1/|c|^2)B^TB$ and $Y_0 = Y$, where $Bx \leq c$ is a system of inequalities valid for K, such that both B and c have nonnegative components.

Proposition 4.3 *Constraint (14) with the above choice of S and Y_0 is satisfied by $Y = xx^T$ if x is a 0-1 vector satisfying $Bx \leq c$.*

Proof.

$$Y - Y_0 S Y_0 = xx^T - (1/|c|^2) xx^T B^T B xx^T \quad (21)$$

$$= (1 - (1/|c|^2)|Bx|^2) xx^T \quad (22)$$

$Bx \leq c$ implies $|Bx|^2 \leq |c|^2$, thus the last expression is indeed psd. ■

A simple observation shows that we can use the psd constraint (14) in a restricted way: if in our higher dimensional formulations we do not have all possible variables $y_{i\ell}, 1 \leq i, \ell \leq n$, i.e. we only generated some elements of the $n \times n$ matrix Y, all we must do is enforce psdness of the available principal submatrices.

5 Generating Cutting Planes from Semi-Definite Constraints

In this section we address the question of generating cuts in the x-space, based on the positive semi-definiteness arguments developed in Section 4.

Consider a higher-dimensional representation of our 0–1 program

$$Dx + By \geq c \quad (23)$$

where the linear inequalities arise from Step 2 of Procedures 3 and from the valid constraints for the convex set defined by (14). Note that system (23) has infinitely many constraints. Nevertheless, for the sake of simplicity, we use the above matrix notation.

The components y_{ij} of the vector y can be viewed as elements of a square matrix Y with diagonal x. It was shown in Section 4 that the separation problem for the semi-definite constraints in (23) amounts to computing a minimum eigenvalue. In our computational experiments, we use the following version of the separation algorithm.

1. Given $(\bar{x}, \bar{Y})$, find the smallest eigenvalue of $\bar{Y} - \bar{x}\bar{x}^T$. If it is nonnegative, $\bar{Y} - \bar{x}\bar{x}^T \succeq 0$. Otherwise, let $v \in \Re^n$ be a corresponding nonzero eigenvector.

2. An inequality of (23) violated by $(\bar{x}, \bar{Y})$ is $(Y - \bar{x}\bar{x}^T) \circ vv^T \geq 0$.

Suppose that we would like to generate a cutting plane $\alpha x \geq \beta$ in the space of the x variables from system (23). In the rest of the section we show that the separation algorithm will suffice to generate the columns of our cut-generation LP; as a by-product, we also get a nice interpretation of its dual.

Assume that we are given a current fractional point, $\bar{x}$. To derive a cutting plane of the form $\alpha x \geq \beta$ (with $\alpha \leq 0$ and $\beta < 0$) which cuts off $\bar{x}$ by the largest amount, we must solve

$$\begin{array}{lllll} max & -\alpha\bar{x} + \beta & & & \\ s.t. & \alpha - uD & = & 0 & \\ & uB & = & 0 & \\ & -uc + \beta & \leq & 0 & \\ & u \geq 0. & & & \end{array} \tag{24}$$

Suppose that we normalize our system by choosing $\beta = -1$. Next, α can be eliminated from (24) to get

$$\begin{array}{lllll} max & u(-D\bar{x}) & & & \\ s.t. & uB & = & 0 & \\ & -uc & \leq & 1 & \\ & u & \geq & 0. & \end{array} \tag{25}$$

Consider the dual of (25),

$$\begin{array}{lllll} min & t & & & \\ s.t. & t & \geq & 0 & \\ & By - ct & \geq & -D\bar{x} & \end{array} \tag{26}$$

which can be rewritten as

$$\begin{array}{lllll} min & t & & & \\ s.t. & t & \geq & 0 & \\ & D\bar{x} + By & \geq & ct. & \end{array} \tag{27}$$

Comparing (23) and (27), we can interpret (27) as follows: Since we are able to generate a cut violated by $\bar{x}$, there cannot be a y with $(\bar{x}, y)$ satisfying (23). Hence the system $D\bar{x} + By \geq ct$ is infeasible for $t = 1$, and the minimum of t in (27) is attained for some $t^* > 1$. The larger the value of t^*, the larger the amount $-(\alpha\bar{x} + 1)$ by which the point $\bar{x}$ is cut off by the inequality $\alpha x \geq -1$. Thus we have the following duality relationship:

Theorem 5.1 { *The maximum amount by which $\alpha x \geq -1$ cuts off $\bar{x}$ equals 1 plus the minimum value of t for which the system $A\bar{x} + By \geq ct$ becomes feasible.*

Also, (27) makes it clear how to generate a cut, even if (23) is only given by a separation oracle. Finding a column with negative reduced cost in (25) can be done by constructing a dual vector (t, y), then checking whether (t, y) satisfies the constraints of (26). This can be done using the same separation algorithm as for (23).

6 Computational Experience

The purpose of this section is twofold: First, we want to investigate the strength of our various cuts, in particular those obtained by using positive semi-definiteness. Second, we want to see how a general-purpose mixed integer programing code, using Procedure 1 in a branch-and-cut framework, performs on the maximum clique problem. We report on our computational experience with several instances from the testbed provided in the DIMACS Challenge.

All experiments were performed starting from formulations $FC(G)$ or $SC(G)$ described in Section 1. The stable sets S_k, $k = 1, \ldots, m$, covering the edges of G in the definition of $SC(G)$ are generated using a greedy procedure. In the generic step of the procedure, called GENSTABLE, we have a family $\mathcal{S} = \{S_1, \ldots, S_{k-1}\}$ of stable sets. If $(i, j) \notin E$ and no stable set of $\mathcal{S}$ contains both i and j, initialize $S_k = \{i, j\}$. Among the vertices of G which have no neighbor in S_k, select one with lowest degree in G, add it to S_k and repeat until S_k is a maximal stable set. Add S_k to $\mathcal{S}$.

6.1 Comparison of the cuts

Before comparing alternatives for generating cuts, we first give a feel for the strength of the higher dimensional relaxations $M_j(K)$, $H_S(K)$ and $H_{1\ldots k}(K)$. We illustrate the difference on the test problem hamming6-4.

The gap closed by Procedure 1 was $\frac{6.28-6.21}{6.28-4} \approx 3\%$, that closed by Procedure 2 ($|S| = 4$) was $\frac{6.28-6.08}{6.28-4} \approx 9\%$ and that closed by Procedure 3 ($|T| = 4$) was $\frac{6.28-5.88}{6.28-4} \approx 17\%$. In all cases the choice of variables was based on their fractionality: we observed (and it is supported by

theoretical evidence [4], page 94) that using components of the current solution which are close to 1/2 usually gives the greatest improvements in Procedures 1-3. This example shows that substantial gains can be achieved by using the stronger formulations, and Procedure 3 seems the most attractive.

We conducted three experiments with Procedure 3 to generate cutting planes. In all cases we chose $k = 6$, $S_j = \{i_j\}$, $j = 1, ..., k$ and $T = \{i_1, ..., i_k\}$. That is, each of the k stable sets we multiplied by to obtain our higher dimensional representation, consisted of a single vertex. The above choice of k is justified as follows: it should be large enough, so that positive semi-definiteness can make a difference; and small enough so that the cut-generation linear programs can still be solved in a reasonable amount of time.

Our test set contained six randomly generated graphs, three with 30 vertices and 20, 30 and 40 % density, respectively, and three with 50 vertices and 15, 20 and 25 % density.

Our first experiment was done without using semi-definiteness in the cut-generation. When we used $FC(G)$ as our initial relaxation, in every case we obtained a cut that could be obtained with Procedure 1 with some $\{i_j\}$ for $j \in \{1, ..., k\}$. These results are not surprising: As mentioned in Section 2, when $K = FC(G)$,

$$N_T(K) = \cap_{j \in T} conv(K \cap \{x : x_j \in \{0, 1\}\}).$$

Since our procedure generates facets of $N_T(K)$ and the facets of $N_T(K)$ are facets of $conv(K \cap \{x : x_j \in \{0,1\}\})$ for some $j \in \{1, ..., k\}$, they are also obtainable by using Procedure 1. When we used $SC(G)$ as our initial relaxation, we still observed the same behavior in most cases.

In our second experiment we used semi-definiteness. First, we solved our initial LP-relaxation. Then we chose our 6 variables T to be as fractional as possible, and in a way that the corresponding vertices induced a *dense subgraph.* Our experience was that semi-definiteness did not improve our cuts in any of the cases, no matter whether our initial relaxation was $FC(G)$ or $SC(G)$.

In our third experiment we also used semi-definiteness. Again, first we solved the initial LP-relaxation. Then our 6 variables T were chosen to be as fractional as possible, and such that the corresponding variables induced a *sparse subgraph.* Our experience is that semi-definiteness substantially improved our cuts, when our initial relaxation was $FC(G)$. In Table 1 we show the improvement in the depth of the generated cuts, averaged over 4 cuts,

Vertices	Density	Depth without PSD	Depth with PSD
30	20	0.5	0.516
30	30	0.5	0.791
30	40	0.5	1.0
50	15	0.5	0.58
50	20	0.5	0.732
50	25	0.5	0.903

Table 1: Improvement in cut-depth using positive semi-definiteness

when our initial relaxation was $FC(G)$. (The *depth* of a cut $\alpha x \geq \beta$ is the euclidean distance between the point $\bar{x}$ it is designed to cut off, and the hyperplane $\alpha x = \beta$). With our initial relaxation chosen as $SC(G)$, the improvements were marginal, or nil.

We summarize the results of this section as follows. Substantial gains can be achieved by using tightening Procedure 3 as compared to Procedure 1, as is examplified by our results on hamming6-4. Unfortunately, these encouraging results do not translate into substantially better cuts. In other words, even though the set $N_T(K)$ is a tighter relaxation than $P_j(K)$, the facets of $N_T(K)$ are often not better than those of $P_j(K)$ when used individually as cuts. The situation improves somewhat with positive semi-definite cuts but not enough to justify the substantial extra computing time. As a consequence, we settled on the cuts generated from Procedure 1 for our more extensive computational study.

6.2 Results with Procedure 1

The lift-and-project cutting planes associated with Procedure 1 are incorporated into a branch-and-cut algorithm. The resulting code, called MIPO, can solve general mixed 0-1 programs, and is described in [2]. In order to apply it to the maximum clique problem we first use the heuristic GENSTABLE described earlier to generate a cover of the nonedges by stable sets. Then we apply MIPO to this formulation. The results are shown in Table 2. Times are in seconds on an Apollo desktop workstation HP720.

The following observations can be made from this table.

For the instances which have a large maximum clique, MIPO did relatively well. Specifically, for the nine instances in Table 2 which have a maximum clique of size greater than 40, only two instance took more than 800 seconds. On the other seven instances, MIPO only took an average of 210 seconds. This is in contrast to the behavior of several other exact algorithms, which tend to deteriorate when the size of the maximum clique increases.

In 40 % of the instances, GENSTABLE generated a formulation with no integrality gap, resulting in excellent computing times. For the problems where cuts were required, the computing times were significantly greater. But remember that MIPO is a general purpose mixed integer program optimizer. We did not taylor it in any way for the maximum clique problem.

One aspect of this class of integer programs is that LP-relaxations tend to have highly fractional solutions. As a result, we can rarely take advantage of the lifting step described in [2], where the cuts are generated in the space of fractional variables of the current solution, then lifted into the whole space.

It is clear from Table 2 that the cuts are working: only five problems required more than 300 cuts and nodes in the branch-and-cut tree. The main drawback is the computing time required to generate the cuts. Thus there is a need to address the question of finding faster cut-generation heuristics. An attractive direction for future research would be to generate cutting planes in subspaces of variables defined by relatively small subsets of vertices, instead of the full space as is currently the case with MIPO.

Problem name	Vertices	Edges	Density	Clique Size	MIPO Cuts	MIPO Branches	CPU Time
brock200-2	200	9876	50 %	12	5140	8746	28000
c-fat200-1	200	1534	8 %	12	120	80	730
c-fat200-2	200	3235	16 %	24	20	2	370
c-fat200-5	200	8473	43 %	58	60	46	2000
C125.9	125	6975	90 %	34	1640	3244	2400
gen200-p0.9-44	200	17910	90 %	44	174	240	670
gen200-p0.9-55	200	17910	90 %	55	0	0	10
hamming6-2	64	1824	90 %	32	0	0	0.13
hamming6-4	64	704	35 %	4	75	54	16
hamming8-2	256	31616	97 %	128	0	0	1.5
hamming8-4	256	20864	64 %	16	78	100	1500
johnson8-2-4	28	210	56 %	4	0	0	0.01
johnson8-4-4	70	1855	77 %	14	0	0	0.15
johnson16-2-4	120	5460	76 %	8	0	0	0.09
keller4	171	9435	65 %	11	2398	4104	6100
p-hat300-1	300	10933	24 %	8	4921	9466	49000
rand100.90	100	4455	90 %	35	56	72	27
rand100.95	100	4703	95 %	52	0	0	0.14
rand200.95	200	18905	95 %	70	2889	4822	7500
san200-0.7-1	200	13930	70 %	30	0	0	11
san200-0.7-2	200	13930	70 %	18	56	38	280
san200-0.9-1	200	17910	90 %	70	0	0	2.6
san200-0.9-2	200	17910	90 %	60	0	0	7.8
san200-0.9-3	200	17910	90 %	44	279	246	780
san400-0.5-1	400	39900	50 %	13	39	18	4400
san400-0.7-2	400	55860	70 %	30	216	162	26000

Table 2: Runs obtained using the cuts of Procedure 1

References

[1] E. Balas, S. Ceria and G. Cornuéjols, A lift-and-project cutting plane algorithm for mixed 0-1 programs, *Mathematical Programming 58* (1993) 295-324.

[2] E. Balas, S. Ceria and G. Cornuéjols, Lift-and-project in a branch-and-cut framework, GSIA working paper (1994).

[3] E. Balas and R. Jeroslow, Strengthening cuts for mixed integer program, *European Journal of Operations Research 4* (1980) 224-234.

[4] S. Ceria, Lift-and-project methods for mixed 0-1 programs, PhD dissertation, Carnegie Mellon Unversity (1993).

[5] M. Grötschel, L. Lovász and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin (1988).

[6] L. Lovász and A. Schrijver, Cones of matrices and set-functions and 0–1 optimization, *SIAM J. Optimization 1* (1991) 166-190.

[7] H. Sherali and W. Adams, A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems, *SIAM Journal on Discrete Mathematics 3* (1990) 411-430.